



**Universitat Autònoma  
de Barcelona**

# **POKERAN: Juego de poker**

Memoria del proyecto  
de Ingeniería Técnica en  
Informática de Gestión  
realizado por

David Molina Catalán

y dirigido por

Salvador Alcántara Cano

Escuela de Ingeniería  
Sabadell, Junio de 2012

El abajo firmante, Salvador Alcántara Cano,  
profesor de la Escuela de Ingeniería de la UAB,

**CERTIFICA:**

Que el trabajo al que corresponde la presente memoria  
ha estado realizado bajo su dirección por

**David Molina Catalán**

Y para que conste firma la presente.

Sabadell, Junio de 2012

-----  
Firmado: Salvador Alcántara Cano

# **Hoja Resumen – Proyecto final de carrera de la Escuela de Ingeniería.**

**Título del proyecto:** PokerAn – Juego de poker

**Autor:** David Molina Catalán

**Fecha:** Junio 2012

**Tutor:** Salvador Alcántara Cano

**Titulación:** Ingeniería técnica en Informática de Gestión

## **Palabras claves:**

- Castellano: Android, Poker, Juego.
- Catalán: Android, Poker, Joc.
- Ingles: Android, Poker, Game.

## **Resumen del proyecto:**

### **- Castellano:**

La finalidad de este proyecto es la realización de una aplicación para dispositivos Android. El juego va dirigido a toda persona que quiera aprender o simplemente disfrutar del juego del poker con personas conocidas. Esta aplicación pretende que jugar al poker sea una práctica sencilla para todo el mundo. Se consigue también que sea el propio sistema el encargado de aplicar las normas y que sin necesidad de conexión a internet sea capaz de jugar con mas personas a la vez en un mismo dispositivo. Se ha escogido este sistema por la gran influencia que tiene y su fácil uso, además de posibilidades y facilidades que ofrece su compatibilidad con un lenguaje consolidado como es Java y XML.

### **- Catalán:**

La finalitat d'aquest projecte és la realització d'una aplicació per a dispositius Android. El joc va dirigit a tota persona que vulgui aprendre o simplement gaudir del joc del poker amb persones conegudes. Aquesta aplicació pretén que jugar a poker sigui una pràctica senzilla per a tothom. S'aconsegueix també que sigui el propi sistema l'encarregat d'aplicar les normes i que sense necessitat de connexió a Internet sigui capaç de jugar amb més persones a la vegada en un mateix dispositiu. S'ha escollit aquest sistema per la gran influència que té i el seu fàcil ús, a més de les possibilitats i facilitats que ofereix la seva compatibilitat amb un llenguatge consolidat com es Java i XML.

### **- Inglés:**

The purpose of this project is to create an application for Android devices. The game is aimed at anyone who wants to learn how to play poker or simply at anyone who wants to enjoy playing the game with friends and family. This application intends to show that poker could be a simple practice for everyone. It has already been proved that the system itself would be responsible for properly applying pokers' rules and also making possible to play, without internet connection, with more people through the same device. The Android system was chosen for the great influence and for being easy to use, as well as its possibilities and facilities for its compatibility with the Java and XML language.

# Índice

Índice de ilustraciones.....	5
1. Introducción.....	6
1.1. Motivación y objetivos.....	7
1.2. Reglas básicas.....	7
1.2.1. Valor de las Combinaciones de las cartas.....	9
2. Estudio de viabilidad y planificación.....	10
2.1. Estudio de viabilidad.....	10
2.1.1. Tipologías y palabras claves.....	10
2.1.2. Descripción y objetivos del proyecto.....	10
2.1.3. Definiciones, acrónimos y abreviaciones.....	11
2.1.4. Partes interesadas.....	11
2.1.5. Referencias.....	12
2.1.6. Producto y documentación.....	13
2.1.7. Estudio de la situación actual.....	13
2.1.8. Requisitos del sistema.....	15
2.1.9. Restricciones del sistema y catalogación de requisitos.....	16
2.1.10. Alternativa y selección de la solución.....	17
2.1.11. Conclusión del estudio de viabilidad.....	17
2.2. Planificación del proyecto.....	18
2.2.1. WBS (Work Breakdown Structure).....	18
2.2.2. Análisis de recursos.....	20
2.2.3. Calendario del proyecto.....	21
2.2.3.1. Dependencias.....	21
2.2.3.2. Cuadro de tareas del proyecto y calendario temporal.....	21
2.2.4. Evaluación de riesgos.....	23
2.2.5. Presupuesto.....	25
Estimación del coste de los recursos.....	25
2.2.6. Conclusiones planificación.....	26
3. Diseño e implementación.....	27
3.1. Técnicas o patrones utilizados.....	27
3.1.1. Patrón del mediador.....	27
3.1.2. Patrón de modelo-vista-controlador (MVC).....	28
3.1.3. Observador.....	29
3.1.4. Singleton.....	31
3.2. Diagrama de clases.....	31
3.2.1. Diagrama de clases generalizado.....	32
3.2.2. Diagrama de clases específico.....	33
3.2.2.2. Diagrama de clases del controlador.....	37
3.3. Diagrama de secuencia.....	39
3.4. Diagrama de casos de uso.....	40
3.5. Algoritmos creados y utilizados.....	41
3.5.1. Control en la combinación de manos.....	41
3.5.2. Diagrama o autómata.....	44
4. Multi-jugador offline.....	46
4.2. Interfaz del juego.....	46
6. Conclusión.....	56
6.1. Desviaciones de la planificación y problemas encontrados.....	56
6.2. Ampliaciones y mejoras.....	57
6.3. Valoración personal.....	57
7. Bibliografía.....	59

## Índice de ilustraciones

Figura: Ejemplo de juego online.....	6
Figura: Lógica del sistema.....	14
Figura: Diagrama WBS.....	21
Figura: Diagrama de Gantt.....	25
Figura: Patrón Mediator.....	30
Figura: Patrón MVC.....	31
Figura: Patrón Observador.....	32
Figura: Patrón Singleton.....	33
Figura: Diagrama de clases generalizado.....	34
Figura: Diagrama de clases del modelo.....	35
Figura: Diagrama de clases extras al modelo.....	39
Figura: Diagrama de clases del Controlador.....	40
Figura: Diagrama de secuencia.....	41
Figura: Diagrama de casos de uso.....	43
Figura: Diagrama de formación de manos, Escaleras.....	47
Figura: Diagrama de formación de manos.....	48
Figura: Pantalla principal.....	52
Figura: Pantalla Acerca de.....	53
Figura: Pantallas de Información.....	54
Figura: Valor de las manos.....	54
Figura: Pantalla de creación de partida.....	55
Figura: Pantalla del mediador o espera del jugador.....	56
Figura: Pantalla de apuesta obligatoria.....	57
Figura: Pantallas individual del jugador.....	58
Figura: Pantallas individual del jugador.....	58
Figura: Pantallas con cambio de turno.....	59
Figura: Pantalla del ganador del turno.....	60
Figura: Pantalla del ganador de la partida.....	60

## Índice de tablas

Tabla: Valor de las manos.....	9
Tabla: Stakeholders.....	11
Tabla: Perfil de usuarios.....	12
Tabla: Equipo de proyecto.....	12
Tabla: Personal del sistema.....	15
Tabla: Requisitos funcionales.....	17
Tabla: Requisitos no funcionales.....	17
Tabla: Fases y actividades del proyecto.....	20
Tabla: Milestones.....	21
Tabla: Recursos del proyecto.....	22
Tabla: Tareas del proyecto.....	24
Tabla: Catalogación de riesgos.....	26
Tabla: Plan de contingencia.....	27
Tabla: Estimación de costes del personal.....	27
Tabla: Estimación del coste de los recursos.....	28

## 1. Introducción

Hasta ahora todos los juegos de poker disponibles para los dispositivos con sistema operativo Android requerían de una conexión a internet para poder disfrutar de los mismos. En la mayoría de ocasiones, sólo se podía jugar con personas desconocidas y con un número de monedas virtuales limitadas, obligando incluso a tener que pagar por disponer de más monedas y poder seguir jugando.

Con este proyecto se pretende llevar el poker a personas que no dispongan de una conexión a internet en el dispositivo o simplemente quieran jugar con conocidos en un mismo dispositivo. De esta manera se quiere favorecer el aprendizaje o simplemente mejorar en el juego del poker.



1. Figura: Ejemplo de juego online

## **1.1. Motivación y objetivos**

El proyecto actual quiere mejorar y dar una visión diferente, contando la aplicación con las siguientes características nuevas para el juego del poker:

- Crear un juego de poker en el que poder jugar con usuarios conocidos.
- Poder jugar en cualquier lugar sin necesidad de disponer de una conexión a internet.
- Poder disfrutar del juego distintas personas en un mismo dispositivo con Sistema Operativo Android.

Con este proyecto se pretende acercar a los usuarios que no han jugado nunca o simplemente quieren jugar con personas conocidas a un juego que ofrece un gran entretenimiento de forma gratuita y ilimitada.

## **1.2. Reglas básicas**

El poker es un juego de cartas en el cual se utilizan 52 cartas (sin comodines). Las cartas están distribuidas en 4 familias: tréboles, corazones, diamantes y espadas/picas.

En esta aplicación se utiliza una variante del poker llamado “*Texas Hold'em*” en el que el jugador dispone de 2 cartas propias (las puede ver únicamente el jugador en cuestión) y 5 comunitarias (compartidas por todos los usuarios). El bote lo ganará el jugador con la mejor formación de 5 cartas de la mesa.

El objetivo del poker es ganar el bote (monedas) acumulado por las apuestas de todos los jugadores de la mesa conseguido por la mejor formación de 5 cartas de la mesa.

El juego del poker se distribuye en 5 etapas en las cuales hay una ronda de apuestas. En cada ronda de apuestas en el turno de cada jugador puede realizar una de las siguientes acciones según haya realizado el jugador anterior:

- Igualar: el jugador realiza una apuesta igual que la de los demás jugadores.
- Pasar: el jugador pasa sin apostar pero no abandona la partida.
- Subir: el jugador sube la apuesta de los demás jugadores.
- Retirarse: el jugador abandona la mano.

## **Etapas**

### **Preflop**

Antes de comenzar la ronda de apuestas existen las apuestas obligatorias que son llevadas a cabo por la ciega pequeña (jugador situado a la izquierda del que ha repartido las cartas o también llamado Dealer) y la ciega grande (jugador situado a la izquierda de la ciega pequeña). La ciega grande apuesta obligatoriamente el doble de la apuesta obligatoria que haya apostado la ciega pequeña.

Después de realizar estas apuestas obligatorias se reparten 2 cartas a cada jugador alternando las cartas. Después de la repartición de las cartas, todo jugador que quiera continuar en la partida deberá igualar la apuesta realizada por la ciega grande excepto si un jugador sube la apuesta, que entonces deberán igualar la de dicho jugador.

Cuando todos los jugadores han igualado las apuestas se pasa a la siguiente etapa.

### **Flop**

En esta etapa, al contrario de el preflop, ya no hay apuesta obligatoria lo que quiere decir que todos los jugadores pueden pasar sin apostar y seguirían estando en la partida. Antes de realizar alguna apuesta, el repartidor elimina la primera carta de la baraja y pone las 3 siguientes sobre la mesa.

Después de la colocación de las 3 cartas comunitarias se comienza la ronda de apuestas, empezando por la ciega pequeña y terminando por el Dealer si ningún jugador sube.

### **Turn**











En el Turn, la primera carta de la baraja se elimina igual pero sólo se colocará una carta en la mesa en lugar de tres. La ronda de apuestas se desarrolla de la misma manera.

### **River**

Esta ronda es exactamente igual que el turn, siendo esta la última ronda de apuestas.



### 1.2.1. Valor de las Combinaciones de las cartas

Valor	Jugada	Ejemplo	Combinaciones
1	Escalera Real		4 entre 2.598.960
2	Escalera de Color		40 entre 2.598.960
3	Poker		624 entre 2.598.960
4	Full		3.744 entre 2.598.960
5	Color		5.108 entre 2.598.960
6	Escalera		10.200 entre 2.598.960
7	Trío		54.912 entre 2.598.960
8	Dobles Parejas		123.552 entre 2.598.960
9	Pareja		1.098.240 entre 2.598.960
10	Carta Alta		1.302.540 entre 2.598.960

1. Tabla: Valor de las manos.

## **2. Estudio de viabilidad y planificación**

### **2.1. Estudio de viabilidad**

Este proyecto es un juego de poker que quiere dar una opción diferente de jugar al poker desde dispositivos móviles que utilicen Sistema Operativo Android, ya que todo el mundo puede acceder fácilmente a esta tecnología, haciendo apuestas con dinero virtual totalmente gratuito.

No todos los apartados del juego serán gratuitos. El juego se podrá mantener gracias a la publicidad dentro del juego, donaciones o beneficios futuros que se podrán adquirir mediante la compra de ítems.

#### **2.1.1. Tipologías y palabras claves**

El tipo de proyecto es un videojuego basado en Android con múltiples posibilidades de implementación:

- Juego multi-jugador offline en el que varios jugadores pueden jugar desde el mismo dispositivo ( alternando el móvil ).
- Juego a través de una red bluetooth donde cada jugador juega desde su dispositivo.
- Futuras implementaciones como podrían ser el multi-jugador online o un único jugador jugando contra la IA de la máquina.

Las palabras claves que identifican el proyecto son: Android, poker, videojuego, multi-jugador offline, multi-jugador, bluetooth.

#### **2.1.2. Descripción y objetivos del proyecto**

##### **Descripción**

Los problemas más frecuentes de los juegos de poker que se pueden encontrar en el Market de Android son el uso exclusivo para jugadores online o que éstos juegos no son gratuitos. Existen problemas asociados como que un jugador quiera acceder a una sala y al estar completa no le deje jugar, o directamente que quiera jugar con amigos o

conocidos y esta opción no se le permita. Este es uno de los puntos en los que se quiere desarrollar esta aplicación.

En este sentido tanto el SO como el poker están en un momento de plenitud social y cada vez más personas juegan al poker o se están iniciando en este hobby. Nuestra mayor motivación es llegar a más personas de una forma diferente, amena y en la que no sea necesario estar conectado a Internet consumiendo datos.

## Objetivos

- Hacer llegar el poker al mayor número de personas utilizando Android.
- Aprender e indagar en el SO Android.
- Planificar un proyecto de unas dimensiones apropiadas.
- Comercializar el videojuego utilizando Android Market de Google.

### 2.1.3. Definiciones, acrónimos y abreviaciones.

SO: Sistema Operativo.

App Store: tiendas no oficiales de Android.

Android: SO de Google.

Android Market: Mercado del SO Android, donde se venden, compran, suben, descargan, se pueden comentar y puntuar aplicaciones.

App: aplicaciones.

### 2.1.4. Partes interesadas.

Stakeholders

Nombre	Descripción	Responsabilidad
A	Director del proyecto	Supervisa el trabajo del alumno, evalúa el proyecto, da la aprobación del proyecto, participa en su definición y hace un seguimiento del proyecto.
BB	Usuario experto	Participa en la definición de los requisitos, subministra información y participa en la validación del proyecto.
CC	Programador	Define principalmente los requisitos y funcionalidades del proyecto, a parte del seguimiento del mismo.

2. Tabla: Stakeholders.

#### Perfil de usuarios

Nombre	Perfil	Responsabilidad
U1	Administrador del sistema	Gestiona y controla el sistema, gestiona los usuarios y es el responsable contable.
U2	Usuario experto	Recoge información sobre los usuarios.
U3	Usuario experto no	Consulta información, ayuda al usuario experto, encargado de comunicarse con los clientes y socios.

3. Tabla: Perfil de usuarios.

#### Equipo de proyecto

Nombre	Descripción	Responsabilidad
A	Jefe del proyecto	Define, gestiona, planifica y controla el proyecto.
BB	Analista	Colabora con el jefe del proyecto en el estudio de viabilidad y la planificación. Analiza la aplicación: arquitectura, metodología, especificaciones, estándares. Participa en el diseño y validación.
CC	Programador	Diseña y desarrolla la aplicación de acuerdo con el análisis y planificación prevista. Participa en el proceso de validación y implementación.
DD	Técnico de pruebas	Participa en el diseño de las pruebas internas y externas. Realiza las pruebas y participa en el proceso de control de calidad.
E	Director del proyecto	Supervisa el trabajo del alumno, en algunos casos también puede actuar como stakeholder.

4. Tabla: Equipo de proyecto

### 2.1.5. Referencias.

- **Normativas de proyectos de Ingeniería Técnica.**  
[http://www.uab.cat/Document/541/595/Normativa\\_PFCNovembre2010.pdf](http://www.uab.cat/Document/541/595/Normativa_PFCNovembre2010.pdf)
- **LOPD**  
<https://www.agpd.es/portalweb/canaldocumentacion/legislacion/estatal/index-ides-idphp.php>
- **Microsoft Project**  
<http://www.microsoft.com/project/>
- **Java**  
<http://docs.oracle.com/javase/6/docs/api/>

– **Android**

<http://developer.android.com/guide/index.html>

### **2.1.6. Producto y documentación.**

Se entregara una aplicación informatizada con la licencia correspondiente.

Se entregara un manual de usuario.

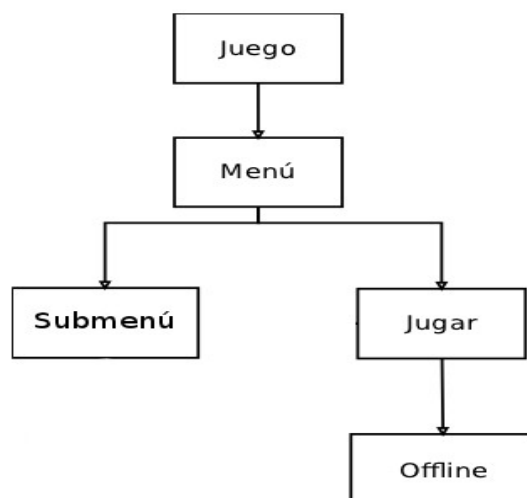
Se elaborará una memoria del proyecto.

### **2.1.7. Estudio de la situación actual.**

La situación actual del juego del poker está en alza, por ese motivo se comenzará con la versión más popular en estos momentos como, “*Texas Hold'em*”, no obstante se incrementaran sus modalidades de juego y también las formas de jugarlo. Para poder gestionar este incremento de jugabilidad se contará con los ingresos por publicidad y donaciones, para así poder dar mejores servicios y prestaciones a los usuarios.

## **Lógica del sistema**

En este diagrama se muestra la lógica principal del sistema.



2. Figura: Lógica del sistema.

## Descripción física

El videojuego podrá ejecutarse sin ningún tipo de problemas en cualquier dispositivo Android con una versión de Android 2.1 a la 2.4, siendo esto ampliable hasta la 4.0.3.

El juego se ha probado en un dispositivo Android con las siguientes características:

- Ram: 512 Mb.
- Procesador: 1 GHz.
- Pantalla: 4”.
- Modelo dispositivo: LG Optimus Black.

Esto no quiere decir que no pueda ser utilizado en otros dispositivos Android, sino que el juego esta optimizado para este dispositivo, o características similares, pudiendo otros de menores o mayores características hacerlo funcionar correctamente.

## Personal del sistema

Nombre	Descripción	Responsabilidad
A	Administrador de usuarios	Encargado de observar, contemplar, registrar que tipo de usuarios descargan o comentan la aplicación para así poder llevar un mayor control sobre los gustos y posibles mejoras a partir de los usuarios.
B	Administrador del programa	Administra y hace el mantenimiento del programa, todo esto manteniendo el contacto diario con el administrador de usuarios.

5. Tabla: Personal del sistema.

## Diagnostico del sistema

\*Deficiencias:

La aplicación actual depende mucho de la publicidad para mantener sus ingresos y poder seguir desarrollando y mejorando la aplicación.

\*Mejoras:

Aumentar los ingresos utilizando las donaciones de los clientes y utilizando micro-pagos por la compra de ítems para el juego, todo esto sin que el juego principal deje de ser gratuito.

## **Normativas y legalización**

### **Normativas necesarias actualmente**

Normativa de proyectos de final de carrera de la Escuela de Ingeniería.

Ley de la propiedad intelectual.

### **Normativas futuras**

LOPD: Ley orgánica sobre la protección de datos. (esta ley no se utilizará en la primera versión del juego, ya que no habrá micro-pagos, ni registros).

## **2.1.8. Requisitos del sistema**

### **Requisitos funcionales**

- (1) Actualización y corrección de la app.
- (2) Gestión de cobro automático de la publicidad.
- (3) Gestión de cobro automático de las donaciones del cliente.
- (4) Control de las descargas y comentarios de los usuarios.

### **Requisitos no funcionales**

- (1) Cumplimiento de la LOPD.
- (2) Los recursos utilizados por la app. han de estar ajustados al dispositivo utilizado.
- (3) Usabilidad. La interfaz ha de cumplir la ISO 9241: Ergonomic requirements for office work with visual display terminals.
- (4) Tolerancia a erratas y acciones incorrectas.
- (5) Control de los errores de usuario.

## 2.1.9. Restricciones del sistema y catalogación de requisitos

### Restricciones del sistema

La aplicación se ha de ejecutar en un entorno Android.

La utilización óptima de la app. se ha desarrollado para un dispositivo concreto, no obstante, utilizable para el resto.

El proyecto ha de estar acabado antes del 26 de Junio de 2012.

### Requisitos funcionales

	RF1	RF2	RF3	RF4
Esencial	X			
Condicional				X
Opcional		X	X	

6. Tabla: Requisitos funcionales.

### Requisitos no funcionales

	RNF1	RNF2	RNF3	RNF4	RNF5
Esencial	X			X	
Condicional			X		
Opcional		X			X

7. Tabla: Requisitos no funcionales.



## **2.1.10. Alternativa y selección de la solución**

### **Alternativa**

*“Live holdem texas pro”*: funciona únicamente utilizando el cobro de fichas por parte de los clientes sin la opción de jugar con publicidad y ser totalmente gratuito.

Funcionalidades:

Salas dedicadas al *“Texas Hold'em”*.

Únicamente tiene la opción de jugar al poker online.

Muchos requerimientos de Hardware para utilizar su software.

### **Solución propuesta**

Nuestra solución con tal de satisfacer a los usuarios es que el juego sea gratuito sin necesidad de pago para poder jugarlo, exceptuando unas características futuras opcionales, que el cliente decide si utilizarlas o no, pero que no afectan al juego ni a su ejecución.

Además de que no afectan al juego del cliente, también tendrá unos ingresos provenientes de la publicidad ingresada en la aplicación.

## **2.1.11. Conclusión del estudio de viabilidad**

En este proyecto los mayores inconvenientes observados son en un principio la inversión inicial y durante los primeros meses el coste de creación y mantenimiento de la app para solucionar los primeros problemas que se puedan detectar por parte de los usuarios. Otro punto a tener en cuenta son los ingresos por parte de la publicidad: menores al inicio por contar con un menor número de usuarios.

Todo lo mencionado anteriormente deberá de amortizarse completamente a partir del segundo trimestre, después del inicio de la aplicación gracias a la publicidad, donaciones de los usuarios, y micro-pagos por ítems extras, la cual comenzará a dar beneficios.

También cabe destacar que al ser un proyecto final de carrera la mayoría de los roles utilizados como personal del sistema se centran en una misma persona con la gran ayuda y orientación del director del proyecto. Por lo tanto este proyecto se amortiza inmediatamente de otra forma que no la económica, como es el aprendizaje de nuevos recursos, implementaciones, lenguajes y planificaciones.

## 2.2. Planificación del proyecto

El plan del proyecto consiste en cuantificar el tiempo y recursos que el proyecto costará. Su finalidad es crear un plan de proyecto que un gestor de proyectos pueda usar para acompañar el progreso del equipo encargado del proyecto.

Este documento recogerá nuestro plan de proyecto, es decir, el conjunto de actividades que nos permitirán desarrollar, ejecutar y controlar el proyecto. El documento incluirá:

- Las tareas y los puntos de control del proyecto.
- Los recursos del proyecto.
- El calendario del proyecto.
- La evaluación de riesgos.
- El presupuesto del proyecto.

### 2.2.1. WBS (Work Breakdown Structure)

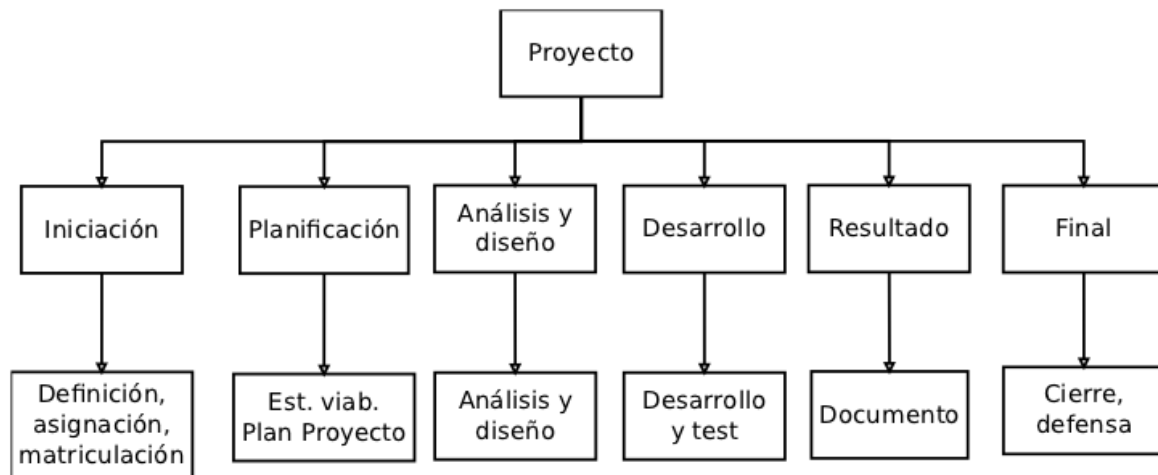
En esta sección veremos todas las fases, actividades y tareas del proyecto.

#### Fases y actividades del proyecto

Fases	Descripción
Iniciación	Incluye las actividades, definición del proyecto, asignación y matriculación.
Planificación	Incluye el estudio de viabilidad y el plan del proyecto.
Análisis	Análisis de requisitos funcionales y no funcionales. Arquitectura del sistema.
Diseño	Incluye el diseño de la capa de datos, del control y la interfaz. Diseño de los test.
Desarrollo	Fase de desarrollo de la aplicación.
Test i pruebas	Fase de prueba del sistema. Incluye test unitarios y de integración.
Implantación	La aplicación se instala en su entorno real.
Generación de documentos	Fase de documentación del proyecto. Incluye manuales y memoria del proyecto.
Cierre del proyecto	Fase de cierre. El director del proyecto firma la aceptación y cierre del proyecto.
Defensa del proyecto	Defensa del proyecto delante de la comisión.

8. Tabla: Fases y actividades del proyecto.

## Diagrama WBS



3. Figura: Diagrama WBS.

## Milestones

En la siguiente tabla podemos ver los puntos de control del proyecto.

Nombre	Descripción	Fecha
Iniciación	Matriculación	01/10/2011
Est. Viabilidad	Aprobación	10/12/2011
Plan del Proyecto	Aprobación	15/12/2011
Análisis	Aprobación	5/03/2012
Diseño	Aprobación	7/04/2012
Cierre	Aceptación	26/6/2012
Defensa	Evaluación	15/7/2012

9. Tabla: Milestones.

### 2.2.2. Análisis de recursos

En este apartado se verán los recursos humanos y materiales asignados para cada tarea.

#### Recursos del proyecto

Recursos humanos	Valoración
Director del proyecto	30€/h
Analista (Alumno)	15€/h
Programador (Alumno)	15€/h
Técnico de pruebas/probador de juegos (Alumno)	15€/h

10. Tabla: Recursos del proyecto.

#### Los recursos humanos utilizados en el proyecto

- Director del proyecto: Iniciación, planificación, soporte de ayuda en la generación de documentos, orientación, cierre y aprobación. Puntos de control.
- Analista: Análisis, diseño, implementación y desarrollo.
- Programador: Diseño, desarrollo y test.
- Técnico de pruebas: Fase de test.

Los recursos humanos se distribuyen en cuatro roles pero realmente identificados en 2 usuarios, el director del proyecto (profesor) y el alumno.

Dp: Director del proyecto.

A: Alumno.

Los recursos materiales utilizados serán los propios disponibles o los prestados por la universidad.

Materiales:

-Teléfono móvil:

- SO: Android 2.1.1.
- Ram: 512 Mb.
- Procesador: 1 GHz.

- PC:

- SO: Ubuntu 11.10.

### **2.2.3. Calendario del proyecto**

- Calendario del proyecto: el proyecto se desarrolla de febrero 2012 a Junio 2012 con una dedicación de unas 5 a 10 horas a la semana. El total de horas dedicadas en el proyecto será de entre 150 y 200 horas.
- Fecha de comienzo: 1 de febrero de 2012.
- Fecha de finalización 20 de junio de 2012.
- Herramientas de planificación y control: Microsoft Project (herramienta de seguimiento y control de desarrollo de proyectos software).

#### **2.2.3.1. Dependencias**

El proyecto se desarrolla linealmente donde cada fase sigue un modelo iterativo hasta conseguir el rendimiento óptimo. Por lo tanto, cada fase no comienza hasta que no se completa la fase anterior.

En la fase de desarrollo se prevé un modelo ágil de tal manera que el diseño, desarrollo y test sigan un modelo iterativo.

La fase de generación de documentos se prevé al final donde se incluirán los documentos elaborados durante el desarrollo del proyecto: inicio, estudio de viabilidad, plan del proyecto, etc.

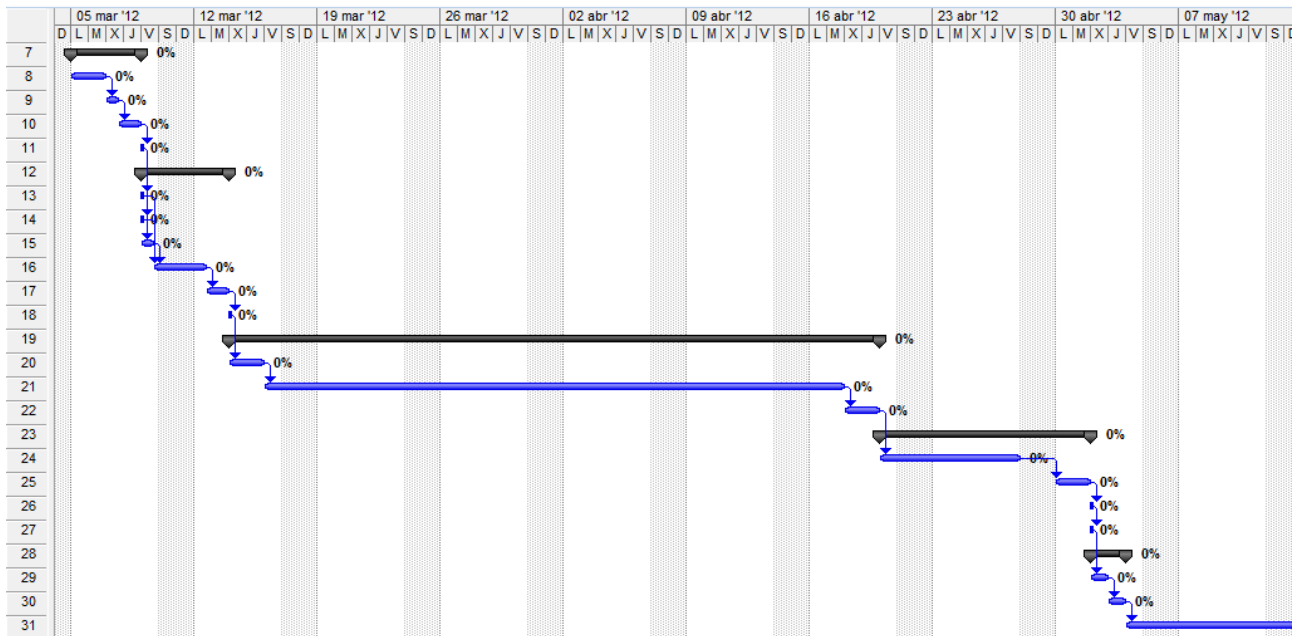
### 2.2.3.2. Cuadro de tareas del proyecto y calendario temporal

En la siguiente tabla podemos observar las tareas y subtareas del proyecto, además podemos encontrar los recursos necesarios para cada una de las tareas y las relaciones entre las tareas.

Nº	Descripción de la actividad	Duración	Recursos	Pred.
1	<b>Inicio del proyecto: asignación y matriculación del proyecto</b>	<b>2h</b>	CP	
2	<b>Planificación del Proyecto</b>	<b>29h</b>		
3	<i>Estudio de viabilidad</i>	12h	CP	1
4	<i>Aprobación EV (Punto de Control)</i>	1h	CP	3
5	<i>Plan del Proyecto</i>	15h	CP	4
6	<i>Aprobación PP (Punto de Control)</i>	1h	CP	5
7	<b>Análisis de la aplicación</b>	<b>20h</b>		
8	<i>Análisis de requisitos (casos de uso)</i>	10h	A	6
9	<i>Análisis de la seguridad y legalidad</i>	1h	A	8
10	<i>Documentación del análisis</i>	3h	A	9
11	<i>Aprobación del análisis (Punto de control)</i>	1h	CP,A	10
12	<b>Diseño de la aplicación</b>	<b>20h</b>		
13	<i>Diseño de la interfaz</i>	3h	A(80%),P(20%)	11
14	<i>Diseño modular de la aplicación</i>	5h	A(80%),P(20%)	11
15	<i>Diseño de la interfaz</i>	6h	A(80%),P(20%)	11
16	<i>Diseño Test de pruebas</i>	2h	A(40%),P(20%),, TP(40%)	13, 14, 15
17	<i>Documentación del Diseño</i>	3h	A(50%), TP(50%)	16
18	<i>Aprobación (Punto de Control)</i>	1h	CP,A	17
19	<b>Desarrollo</b>	<b>125h</b>		
20	<i>Preparación entorno desarrollo</i>	10h	P	18
21	<i>Modulo y funcionalidades de la aplicación</i>	100h	P	20
22	<i>Desarrollo de la interfaz de usuarios</i>	15h	P	21
23	<b>Test y pruebas</b>	<b>34h</b>		
24	<i>Pruebas de integración</i>	24h	P	22
25	<i>Pruebas de estrés</i>	7h	P	24
26	<i>Documentación de desarrollo y test</i>	2h	P	25
27	<i>Aprobación del desarrollo y pruebas (Punto de control)</i>	1h	CP,A,P	26
28	<b>Implantación</b>	<b>8h</b>		
29	<i>Instalación</i>	2h	A(70%),P(30%)	27
30	<i>Pruebas reales</i>	6h	A(60%),P(20%), TP(20%)	29
31	<b>Memoria del proyecto</b>	<b>38h</b>	CP	30
32	<b>Cierre del proyecto</b>	<b>1h</b>	CP	31
34	<b>Defensa del proyecto</b>	<b>3h</b>	CP	33
Nº	Descripción de la actividad	Duración	Recursos	Pred.

11. Tabla: Tareas del proyecto.

## Calendario temporal



4. Figura: Diagrama de Gantt

### 2.2.4. Evaluación de riesgos

Entendemos como riesgo cualquier factor o conjunto de factores que si suceden no permitirán finalizar el proyecto en la fecha establecida.

En el siguiente punto se evaluarán los riesgos.

#### Lista de riesgos

R1. Planificación temporal optimista: Plan de proyecto. No se acaba en la fecha prevista, aumentan los recursos.

R2. Falta alguna tarea necesaria: Plan de proyecto. No se cumplen los objetivos del proyecto.

R3. Cambio de requisitos: Estudio de viabilidad, análisis. Retraso en el desarrollo y resultados.

R4. Herramientas de desarrollo inadecuadas: Desarrollo. Retraso en la finalización del

proyecto, menos calidad, ....

R5. No se hace correctamente la fase de test: Desarrollo, implantación. Falta de calidad, deficiencias en la operativa, insatisfacción de los usuarios, pérdida económica.

R6. Falta de medidas de seguridad: Estudio de viabilidad, análisis, desarrollo. Pérdida de información, incumplimiento legal, pérdidas económicas.

R7. Abandono del proyecto antes de la finalización: En cualquier fase. Pérdida económica, frustración.

## Catalogación de riesgos

En la tabla siguiente se muestra la probabilidad y el impacto de cada riesgo:

	Probabilidad	Impacto
<b>R1</b>	Alta	Crítico
<b>R2</b>	Alta	Crítico
<b>R3</b>	Alta	Marginal
<b>R4</b>	Baja	Crítico
<b>R5</b>	Alta	Crítico
<b>R6</b>	Alta	Crítico
<b>R7</b>	Media	Catastrófico

*12. Tabla: Catalogación de riesgos.*

## Plan de contingencia

En la tabla siguiente se muestran las posibles soluciones de cada riesgo.

Riesgo	Solución
<b>R1</b>	Retrasar alguna funcionalidad, afrontar posibles perdidas, hacer un seguro.
<b>R2</b>	Revisar el plan del proyecto, modificar la planificación.
<b>R3</b>	Renegociar con el cliente, retrasar funcionalidad, modificar planificación y presupuesto.
<b>R4</b>	Prever herramientas alternativas, mejorar la calidad.
<b>R5</b>	Diseñar los test con antelación, realizar test automáticos, negociar contrato de mantenimiento, dar garantías, afrontar pérdidas económicas.
<b>R6</b>	Revisar la seguridad de cada fase, aplicar políticas de seguridad activas.
<b>R7</b>	No tiene solución.

*13. Tabla: Plan de contingencia.*



## 2.2.5. Presupuesto

A continuación podemos ver la solución propuesta de acuerdo con la planificación del proyecto que se ha hecho anteriormente.

### Estimación del coste del personal

En la tabla siguiente se muestra la estimación del coste de personal asignado en el proyecto:

Personal	Duración total	Coste total
Director del proyecto	76,00 h	2280,00 €
Analista	40,50 h	607,50 €
Programador	164,00 h	2460,00 €
Técnico de pruebas	3,10 h	46,00 €
<b>Total</b>	<b>283,60 h</b>	<b>5393,50 €</b>

14. Tabla: Estimación de costes del personal.

### Estimación del coste de los recursos

En la tabla siguiente se verán los costes de recursos asignados al proyecto:

Material	Coste
Pc	400€
Dispositivo Android	200€
<b>TOTAL</b>	<b>600€</b>

15. Tabla: Estimación del coste de los recursos

### Resumen y análisis del coste beneficio

Calculamos el coste final del proyecto:

Coste de desarrollo .....5393,50 €  
Coste amortización del material.....600,00 €  
TOTAL: 5.993,50 €

Aunque el coste del proyecto es elevado, el coste de material del proyecto se puede amortizar con la utilización en otros proyectos, los ingresos publicitarios y los donativos. También comentar que los costes de material son aproximados ya que en realidad no se ha hecho dicho gasto al ya tener acceso a los materiales utilizados.

### **2.2.6. Conclusiones planificación**

- Se han de determinar las fases, actividades principales y puntos de control del proyecto.
- Se han de representar gráficamente utilizando WBS.
- Se han de valorar los recursos del proyecto.
- Se ha de generar el calendario del proyecto, incluyendo el diagrama de Gantt.
- Se han de evaluar los riesgos del proyecto y preparar un plan de contingencia.
- Se ha de determinar el presupuesto del proyecto.
- Se ha de analizar el coste del proyecto en relación con los beneficios esperados.

### **3. Diseño e implementación**

El propósito de esta sección es definir las funciones del sistema utilizando herramientas como los diagramas de clases, casos de uso, secuencia, etc. Además de utilizar las herramientas anteriormente comentadas, se explicará también el uso de los patrones de diseño utilizados en el proyecto así como los problemas y soluciones que han podido surgir mientras se implementaba.

#### ***3.1. Técnicas o patrones utilizados***

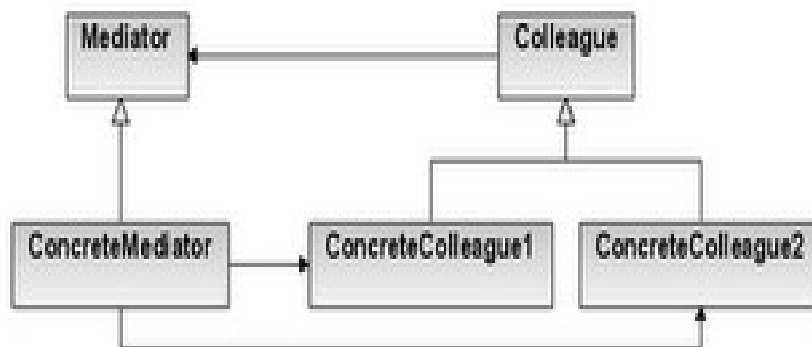
En este proyecto se han tenido que utilizar distintos patrones de diseño para poder llevarlo a cabo. Es un juego para un dispositivo móvil en un sistema nuevo y no conocido por los desarrolladores. Todo esto ha provocado un proceso complejo al tener que desarrollar por primera vez una aplicación con interfaz gráfica y estilo diferente a lo anteriormente desarrollado en la carrera.

##### ***3.1.1. Patrón del mediador***

Este patrón fue diseñado para poder encapsular como interactúan un conjunto de objetos. Promueve un bajo acoplamiento al evitar que los objetos se comuniquen los unos a los otros explícitamente y permite variar la interacción entre ellos de forma independiente.

La intención de este patrón es simplificar la comunicación entre los objetos de un sistema introduciendo un único objeto (mediador) que gestiona la distribución de los mensajes entre los otros (jugadores).

Este diseño se encuentra insertado en nuestro proyecto a la hora de que los jugadores puedan comunicarse entre ellos sin necesidad de hacerlo directamente, es el mediador el encargado de crear los turnos y distribuir a los jugadores.



5. Figura: Patrón Mediator.

En el siguiente patrón se hablara también del patrón mediador.

### 3.1.2. Patrón de modelo-vista-controlador (MVC)

Posiblemente este patrón haya sido el más difícil de implementar por el sistema donde se realizó y al ser la primera vez que se utilizaba, pero a su vez a sido el más útil.

Este patrón se utiliza para separar la lógica del juego y la interfaz de usuario, lo que permite y facilita la evolución por separado de ambos aspectos, además de incrementar su reutilización y flexibilidad del código.

El MVC se distribuye en tres partes: modelo, vista y controlador.

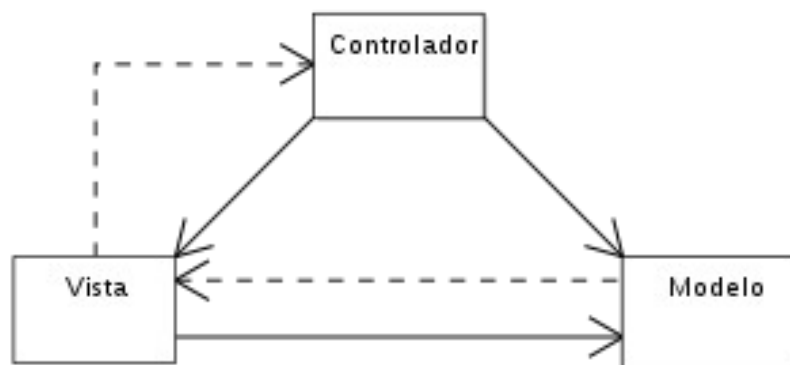
- Modelo: Es el código de la aplicación, las reglas, métodos y clases que lo hacen funcionar.
- Vista: Es el conjunto de interfaces gráficas desarrolladas para el correcto funcionamiento del juego, todas ellas visuales y generadas para interactuar con el usuario.
- Controlador: Es parte del código del juego, con la diferencia de que esta parte se encarga de controlar al modelo y a la vista, para poder incrementar su reutilización y flexibilidad.

En este caso, llevado al terreno de un juego en Android, la vista se ha implementado en formato XML (Extensible Markup Language), lenguaje utilizado por Android para crear interfaces gráficas. No obstante también se puede utilizar Java para crear dicha interfaz con el inconveniente de que el controlador y la vista se deberían de unificar, la cual cosa no deseamos para poder tener de esta forma una separación más visible de los tres elementos de este patrón.

El Controlador está implementado en lenguaje Java igual que el modelo, a diferencia del modelo que aporta las funciones básicas del juego. El controlador las unifica para poder así crear un funcionamiento uniforme del juego y poder comunicarse con la vista para poder interactuar con los usuarios. En este caso el controlador incorpora el patrón del mediador, para poder comunicar de esta forma los datos o modificaciones a los usuarios activos del juego.

El modelo del juego en si es el código en Java con las funciones y clases que hacen posible el correcto funcionamiento de la aplicación.

Este patrón puede variar su implementación dependiendo del lenguaje de programación.

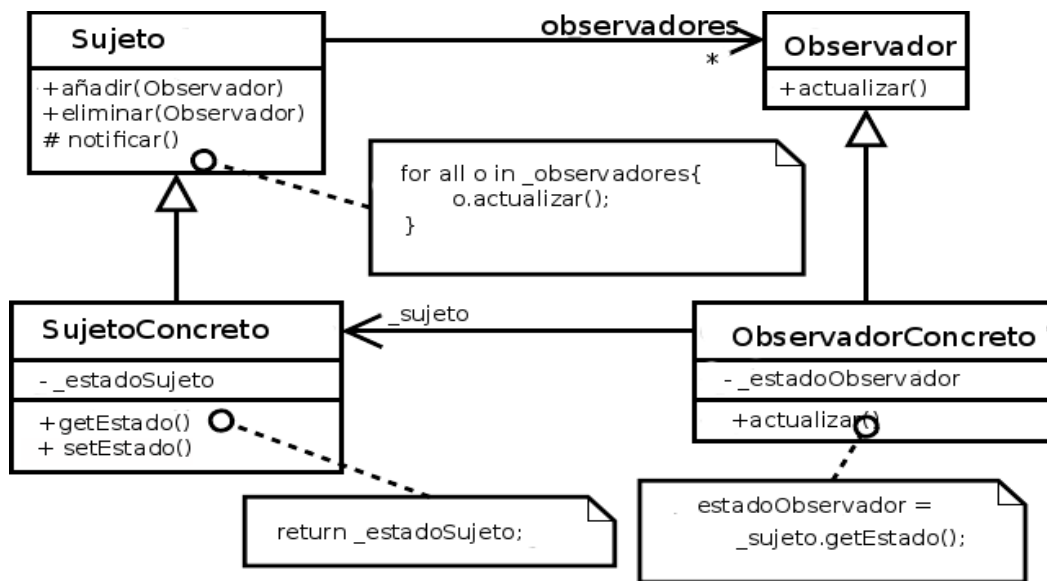


6. Figura: Patrón MVC.

### 3.1.3. Observador

Este patrón de diseño define la dependencia de un tipo de objetos a muchos, de manera que cuando uno de los objetos cambia su estado, éste debe notificar el cambio a los objetos que dependen de su cambio. Este patrón de comportamiento no solo describe la relación entre las clases y objetos sino también los esquemas de comportamiento entre ellos.

El funcionamiento de este patrón es el siguiente:



7. Figura: Patrón Observador.

Los objetos de la clase “observador” quieren saber cuando cambian los datos que muestran el “sujeto observado” para poder actualizarse.

Lo bueno de estas clases, observadora y observada, es que no dependen la una de la otra, lo que quiere decir que las clases no saben de su existencia, ya que este motivo reduciría su reutilización. A fin de cuentas, todas las clases observadoras están en contacto indirectamente con la clase observada, la cual avisa de los cambios a los objetos que la observan.

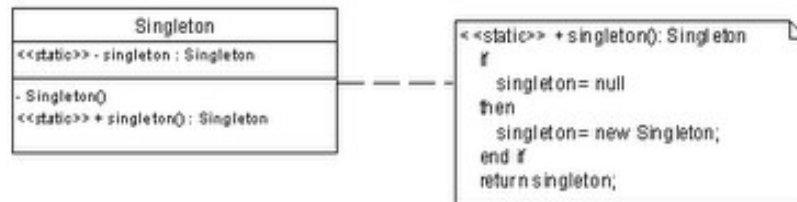
En nuestro proyecto este patrón no se utiliza en una labor muy importante, pero si ahorra recursos del sistema, ya que se utiliza en el consultor de tiempo para las jugadas de los usuarios, para que así no excedan su tiempo límite. Al crear el este patrón solo se crearía un objeto de la clase reloj y los objetos “Player” del modo multi-jugador offline solo tendrían que observar un único reloj y no tener que crear tantos relojes como jugadores.

### 3.1.4. Singleton

Este patrón es muy utilizado en los lenguajes OO y sobre todo en java por el siguiente motivo: En Java cada vez que se crea un objeto nuevo se invoca al constructor del objeto, para crear una nueva instancia (por lo general los constructores en java son públicos). El Singleton lo que hace es convertir el constructor en privado, de manera que ningún lo pueda instanciar.

El objeto se puede instanciar a partir de un método público y estático de la clase. El método revisa si el objeto ha sido instanciado antes y si no lo ha sido, llama al constructor

y guarda el objeto creado en una variable estática del objeto. Si el método ya se ha instanciado anteriormente, lo que hace el método es devolver la referencia a la variable creada anteriormente.



8. Figura: Patrón Singleton.

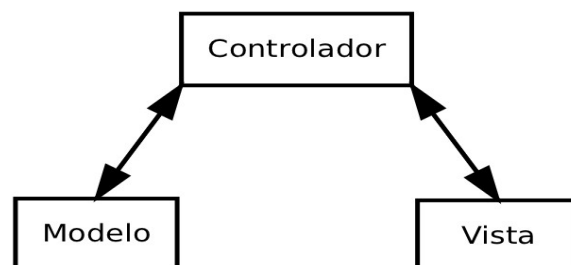
Este patrón se utiliza en el proyecto en concepto de la identificación de los jugadores para gestionar las conexiones entre los usuarios en el modo multi-jugador offline.

## 3.2. Diagrama de clases

En este apartado se mostrarán y explicarán las clases generales, en su agrupación general partiendo del patrón MVC, y el estado más específico del diseño del programa y sus clases, incluyendo los métodos más importantes de estas clases.

### 3.2.1. Diagrama de clases generalizado

En este diagrama podemos observar que el controlador es el único que conoce la existencia de las otras dos clases.



9. Figura: Diagrama de clases generalizado.

El conjunto de clases Controlador utiliza las funciones creadas por el modelo y hace que los resultados se ejecuten en el conjunto de vistas haciendo llamadas a ellas mismas y modificando sus parámetros.

El modelo en si solo tiene información propia, variables que él utiliza y sus propios métodos.

Al conjunto de vistas las controla el controlador, ella misma no sabe cual de sus vistas va a ser visible para el usuario o cual está en ejecución. Esta tarea recae completamente en el controlador para así poder pasar parámetros, variables y todos los cambios efectuados al modelo.

A continuación se especifican los detalles de estos tres grupos de clases.

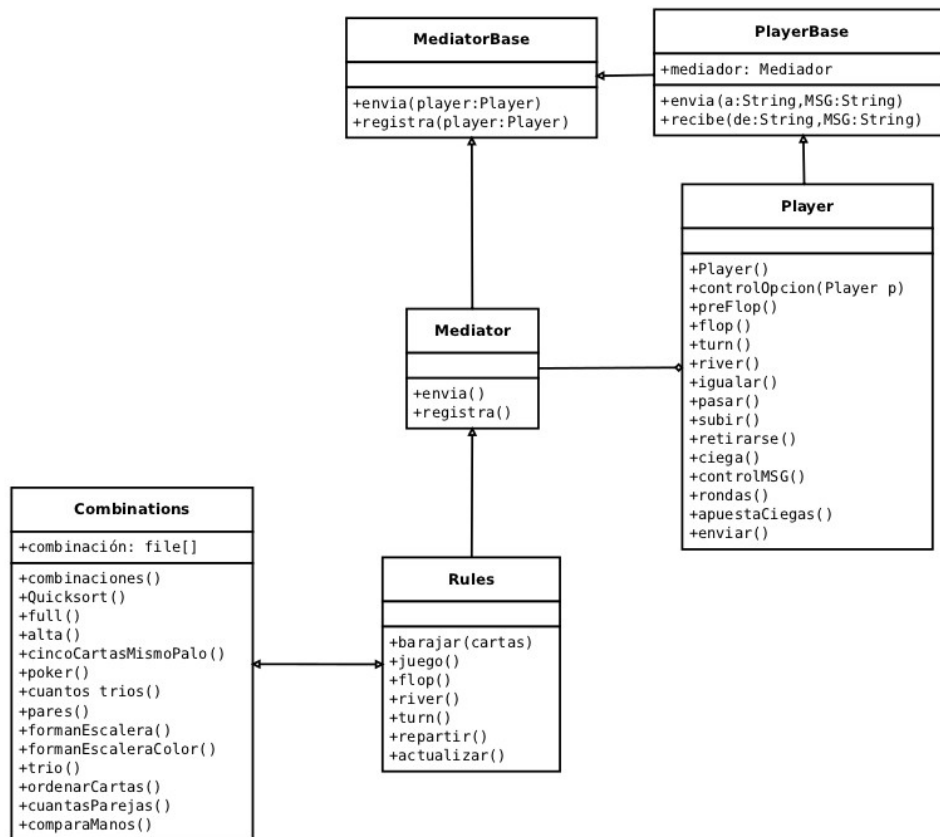
### ***3.2.2. Diagrama de clases específico***

En este apartado se verán y explicarán el funcionamiento y diseño de los grupos de clases anteriormente mostrados.

#### **Diagrama de clases del modelo**

A continuación podemos observar las clase principales que forman el modelo que se explicarán al observar el diagrama.





10. *Figura: Diagrama de clases del modelo.*

Explicación detallada de cada diagrama con los métodos principales:

MediatorBase: es una clase abstracta en la que se indica los métodos enviar y registrar utilizados posteriormente por el Mediator.

PlayerBase: es una clase abstracta en la que se indica los métodos enviar y registrar implementados posteriormente en la clase hijo Player.

## Mediator

Clase pública encargada principalmente de definir los argumentos utilizados después por las posibles clases que hereden sus métodos. También implementa los métodos de la clase abstracta padre:

- enviar(): se encarga de enviar un mensaje a la clase Player para decirle que es su turno.
- Registra(): este método registra a los Player para que se puedan comunicar entre ellos.

## Rules

Esta clase es la encargada de hacer cumplir las reglas principales y mediar en el juego entre los objetos Player registrados.

- `barajar()`: este método se encarga de barajar las cartas al inicio de cada juego.
- `repartir()`: se encarga de repartir las cartas a todos los jugadores registrados en el Mediator.
- `actualizar()`: al inicio del juego se llama a este método para que se actualicen e inicialicen todas las variables necesarias.
- `flop()`: este método es utilizado para colocar las primeras tres cartas comunitarias encima de la mesa, y que todos los Player las puedan ver. Se coloca en la fase del flop.
- `turn()`: este método es como el anterior con la variación de que solo coloca una carta encima de la mesa. Se coloca en la fase del turn.
- `river()`: al igual que el método anterior coloca una carta comunitaria encima de la mesa, siendo esta la última carta colocada en el juego. Se coloca en la última fase el river.
- En esta clase también se utiliza la clase “Combinaciones” (que se explicarán más adelante).

## Player

Esta clase pública es la encargada de controlar las opciones disponibles que puede utilizar el usuario y en los distintos turnos que la clase Mediator le ha enviado.

- `Player()`: esta indicación no se refiere a un método si no al constructor de clase en el cual se debe añadir un Mediator, para que los objetos Player con el mismo Mediator se puedan comunicar y tener las mismas variables de juego.
- `preflop()`, `flop()`, `turn()`, `river()`: estos métodos se agrupan ya que su funcionalidad es similar: es controlar las apuestas que los objetos Player hacen en el turno que lleva su nombre.
- `rondas()`: este método agrupa y indica qué método de los anteriormente nombrados debe utilizarse.
- `controlOpcion()`: el siguiente método es el encargado de decirle al objeto Player que opciones de las siguientes puede utilizar en su turno (igualar, pasar, subir, retirarse).
- `igualar()`: esta función indica cuando un jugador iguala la apuesta del anterior

jugador que hizo la apuesta más alta.

- pasar(): este método hace que el Player que la elija pase, lo cual hace que no apueste, pero no abandona la jugada, ya que todos los usuarios que anteriormente han apostado en el turno no han subido todavía y la apuesta general esta igualada.
- subir(): este método indica que el jugador quiere subir la apuesta del anterior jugador que ha apostado.
- retirarse(): hace que automáticamente el jugador que la utiliza no vuelva a jugar durante dure el juego. Una vez el juego comience de nuevo al llegar el turno del preflop el jugador volverá a estar activo y podrá jugar de nuevo.
- ciega(): esta función se llama automáticamente cuando el jugador es ciega grande o pequeña, ya que su apuesta debe ser obligatoria.
- ApuestaCiegas(): este método se llama directamente al llamar a la función ciega() y es la encargada de hacer la apuesta automática.
- controlMSG(): este método hace que se identifique y cree el mensaje que tiene enviar un usuario al siguiente.

## Combinaciones

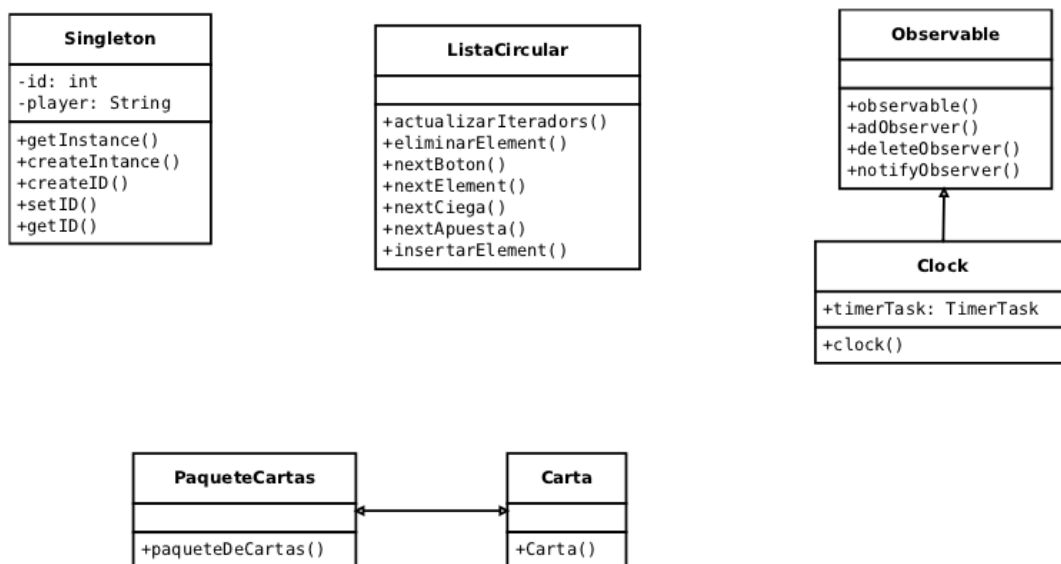
Esta clase se encarga de comparar las combinaciones de manos creadas por los jugadores y decidir cual de ellos ha sido el ganador.

En esta clase se hará mención a un método de ordenación que es el quicksort utilizado en esta clase para ordenar las cartas de mayor a menor para así poder identificar las manos creadas, que, a su vez, se identifican después de la ordenación utilizando los siguientes métodos:

- comparaManos(): este método utiliza los siguientes métodos para comparar las manos de los jugadores. Mezcla las cartas comunitarias con las personales y las ordena para poder saber cuál es la mejor combinació de manos que tiene el jugador.
- combinaciones(): este método indica la mejor mano de un jugador.
- cincoCartasMismoPalo(): dicho método comprueba si hay cinco cartas del mismo color, si las hay llama al método formanEscaleraColor(), si no las hay llama a formanEscalera().
- formanEscaleraColor(): comprueba si hay escalera de color o real y indica cuál de las dos tiene el jugador.
- formanEscalera(): comprueba si hay escalera, si la hay devuelve el resultado al jugador, si no llama a los siguientes métodos.
- alta(): comprueba la carta alta del jugador.

- pares(): mira si hay alguna pareja.
- cuantasParejas(): comprueba si el usuario tiene dobles parejas.
- trio(): se comprueba si hay algún trío.
- cuantosTrios(): este método comprueba si hay más de un trío y indica cuál es el más alto.
- poker(): mira si hay cuatro cartas iguales, es decir, que el jugador tenga poker.
- full(): indica si el jugador tiene full.

A continuación podemos ver el resto de clases del modelo, las cuales intervienen en el modelo pero no como el funcionamiento en si del juego.



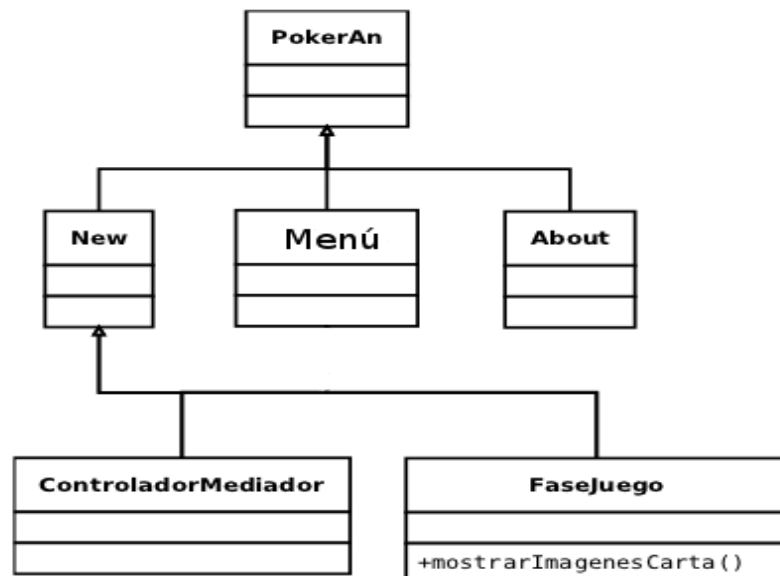
11. Figura: Diagrama de clases extras al modelo.

- La clase Singleton es la encargada de hacer que cada usuario (objeto de la clase Player), tenga una única identificación para el modo multi-jugador por bluetooth.
- La clase ListaCircular hace que la clase Mediator pueda controlar los movimientos entre los jugadores, controlar los Dealer y los movimientos de ciegas.
- La clase clock que utiliza el patrón observer para que los objetos Player puedan observar el tiempo transcurrido para hacer la apuesta en el modo multi-jugador por bluetooth.

- La clase PaqueteCartas y Carta están relacionadas. La clase PaqueteCartas incluye el tipo de cartas que hay, de As hasta el Rey y con sus cuatro palos, creando de esta forma el mazo de 52 cartas que tiene la baraja de poker, y la clase Carta hace referencia individualmente a las cartas en si de la clase anterior.

### 3.2.2.2. Diagrama de clases del controlador

En el siguiente diagrama se pueden ver las clases que forman parte del grupo controlador, encargado de mediar entre las vistas y el modelo.



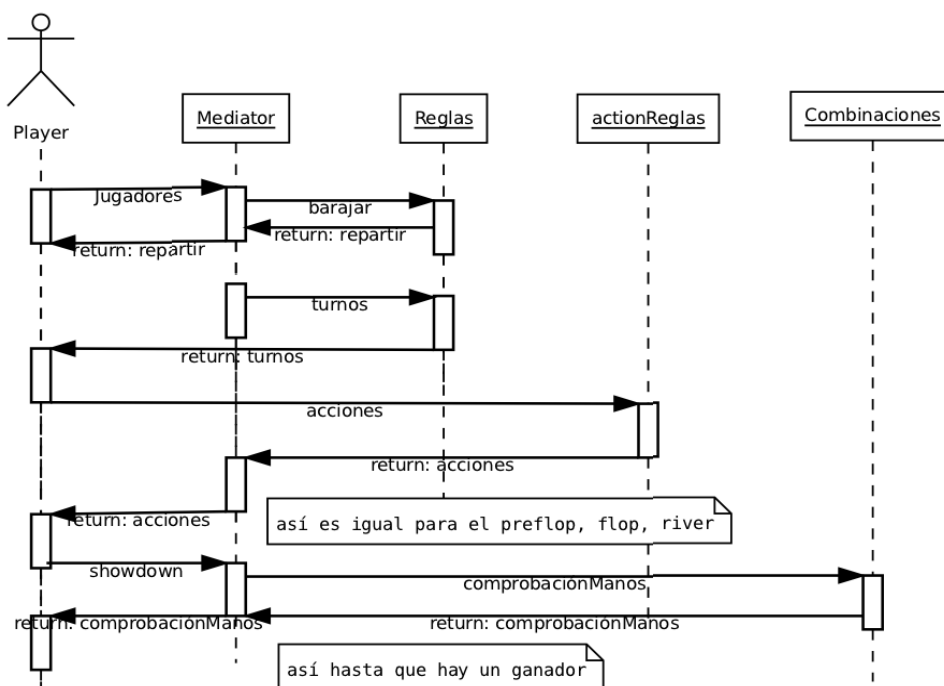
12. Figura: Diagrama de clases del Controlador.

- La clase **PokerAn** del grupo controlador es la encargada de controlar el funcionamiento de la pantalla inicial del juego en la cual llama a su vista **PokerAn.xml** y puede ejecutar acciones que llaman a otros controladores encargados de otras vistas y funciones.
- La clase **New** encargada de crear un nuevo juego creando a los jugadores, llamar a su vista y llamando al controlador mediador para que comience el juego.
- La clase **Continuar** llama directamente a la clase **ControladorMediator** para que se continúe la partida.

- La clase About lanza una vista acerca del creador, director de la aplicación, etc...
- La clase ControladorMediator es la encargada de lanzar la vista controlador\_mediador.xml, encargada de esperar a que el jugador del turno este presente para apostar y llamar a las funciones pertinentes. Acto seguido en la propia clase llama a la clase controladora FaseJuego.
- La clase FaseJuego se encarga de controlar las decisiones del jugador, al igual que mostrar la vista propia (fase\_juego.xml), enviar los datos y llamar a la clase ControladorMediator.

### 3.3. Diagrama de secuencia

A continuación veremos el diagrama de secuencia del funcionamiento de las clases principales.



13. Figura: Diagrama de secuencia.

El diagrama de secuencia comienza con la clase Player que es a su vez controlada por un usuario Actor. Este actor ingresa los usuarios o actores que estarán disponibles para jugar la partida y los registra en el Mediator.

El Mediador llama a la función de la clase Rules barajar. Dicha clase devuelve el paquete de cartas ya barajadas al mediador, al mediador que las reparte a los jugadores en el momento idóneo.

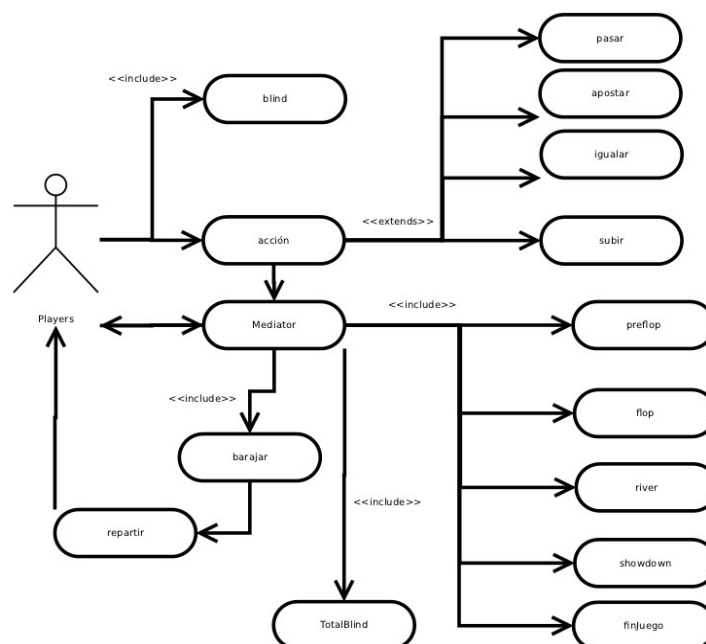
Acto seguido la clase Mediador indica a la clase Player (de un determinado actor) que puede comenzar con su apuesta en el primer turno (preflop). El actor hace su apuesta y pasa el testigo al Mediador de la partida que se encarga de hacer lo mismo con otro actor y así continuar los turnos para cada jugador hasta que se llega al ultimo turno el river y se da por finalizado el turno de apuestas.

Después de finalizar el turno de apuestas el mediador utiliza el método comprobacionManos de la clase Combinaciones que indicará qué usuario a sido el ganador del juego.

Este diagrama de secuencia se repite sistemáticamente hasta que hay un único actor activo en la partida (indica que este jugador ha ganado), se repetirá a partir del punto de que el Mediador indica a la clase Player que puede iniciar la primera apuesta del turno (preflop).

### 3.4. Diagrama de casos de uso

En el siguiente diagrama veremos y explicaremos el diagrama de casos de usos del juego de poker.



14. Figura: Diagrama de casos de uso.

Explicación:

El actor “Player” es el encargado de interactuar con el programa.

Antes de comenzar el juego el “Mediator” reparte las barajas a cada jugador.

Mediator es el proceso principal encargado de distribuir los turnos. El primero, pre-flop, se encarga de organizar las apuestas sin ninguna carta sobre la mesa. El segundo, flop, se encarga de quemar una carta y situar tres cartas comunitarias en la mesa además de organizar apuestas. El turn hace lo mismo que flop pero solo colocando una carta en la mesa comunitaria. El siguiente es el river, encargado también de organizar apuestas, quemar una carta y poner otra más sobre la mesa. Por último esta el showdown que se refiere al acto de comprobar las manos de los jugadores y mostrar el ganador.

Al final de cada Showdown, Mediator se encarga de mirar si ha finalizado el juego y comprobar las combinaciones para ver qué jugador ha ganado la partida actual.

En cada turno el Mediator se encarga de avisar al Player para que tome una decisión sobre las acciones disponibles, controlar el blindTotal y el blind de cada jugador.

### ***3.5. Algoritmos creados y utilizados***

En esta sección se explicarán los algoritmos utilizados y también se mostrará su diagrama correspondiente.

#### ***3.5.1. Control en la combinación de manos***

Este algoritmo se llevó a cabo por la complicación de comparar distintas cartas de distinto valor y que en una combinación específica una mano puede tener un valor u otro.

Este algoritmo se lleva a cabo dentro de la clase Combinations llamando a la función comparaManos().

Antes de comenzar a explicar este algoritmo hay que comentar que hay otro algoritmo dentro de este mismo: el quicksort que es el encargado de ordenar las cartas en su valor numérico. Este valor numérico se asigna a base de la comparación de su nombre de tipo String (caracteres) y se ordena y cambia su posición.

Acto seguido mostramos el algoritmo propiamente dicho para las comparación de las manos de los jugadores:



Algoritmo: Combinaciones.

Variables: booleano control, booleano full, entero escalera, entero pares, entero tríos, entero parejas, entero poker.

escalera := 0

pares := 0

tríos := 0

parejas := 0

poker := 0

full := falso

control := falso

quicksort("cartas del Player") : combinación cartas comunitarias con propias.

control:=cincoCartasMismoPalo() : ¿tiene cinco cartas del mismo palo?

SI control = cierto ENTONCES

escalera := formanEscaleraColor() :¿forma escalera de color?

SI NO

escalera:= formanEscalera() :¿forman escalera normal?

FINSI

EN CASO escalera SEA

ESCALERA\_REAL\_COLOR:

DEVUELVE ESCALERA\_REAL\_COLOR:

FIN

ESCALERA\_COLOR

DEVUELVE ESCALERA\_COLOR

FIN

ESCALERA

DEVUELVE ESCALERA

FIN

COLOR

DEVUELVE COLOR

FIN

FINCASO

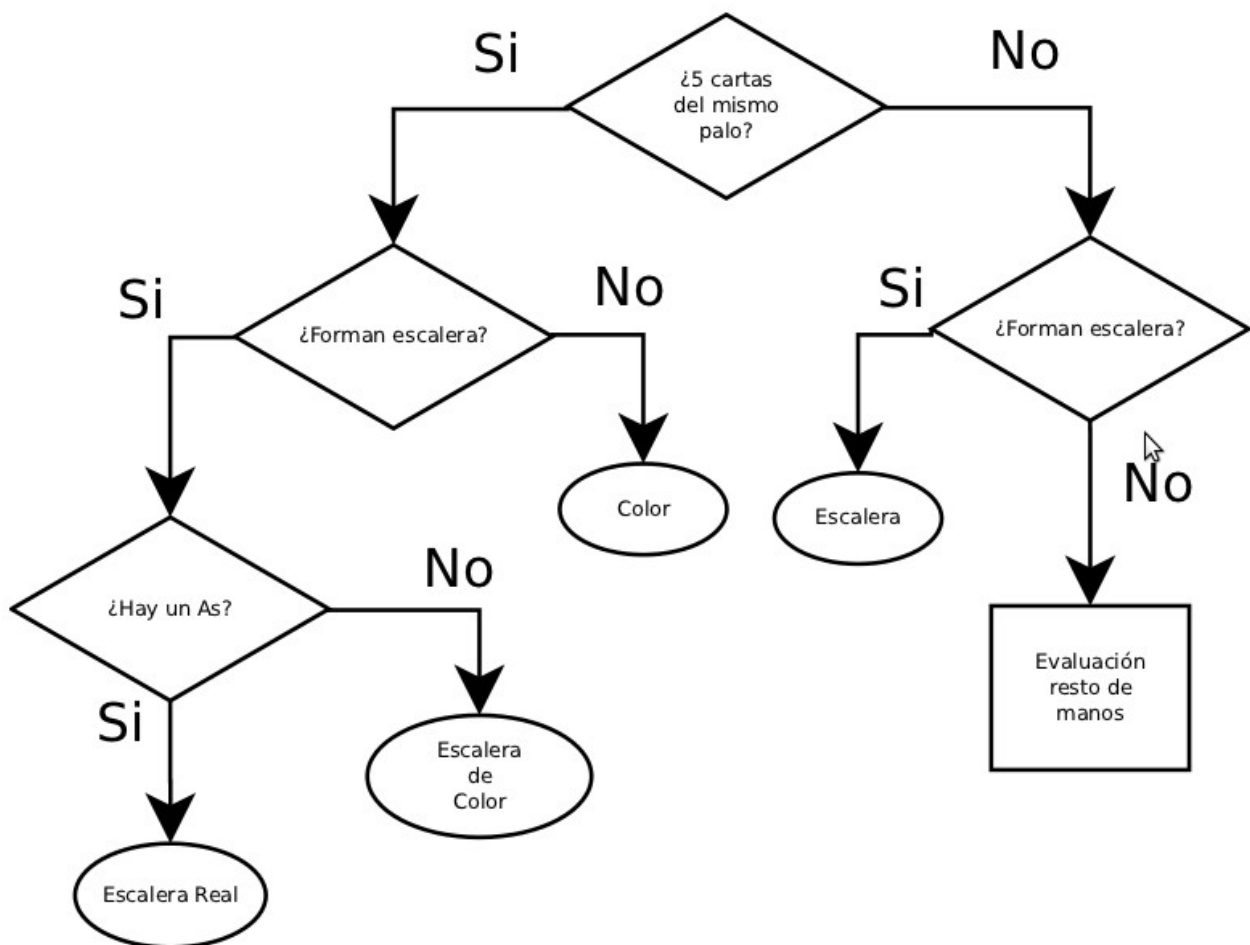
SI escalera = 0 ENTONCES

```
pares = pares()      : ¿hay parejas?
SI pares = CARTA_ALTA ENCONCES
    DEVUELVE CARTA_ALTA
SI NO SI pares = PAREJA ENTONCES
    trío = trío()      : ¿hay algún trío?
FINSI
SI trío != TRIO ENTONCES
    parejas = cuantasParejas()      : mira si hay más de una pareja.
    SI parejas = PAREJA ENTONCES
        DEVUELVE PAREJA
    SINO
        DEVUELVE DOBLE_PAREJA
    FINSI
SINO
    poker = poker()      : mira si hay combinación de poker
    SI poker = POKER ENTONCES
        DEVUELVE POKER
    SINO
        tríos = cuantosTríos()      : comprueba si hay más de un trío.
        SI tríos > 0 ENTONCES
            full = full()      : devuelve si existe full.
            SI full = falso ENTONCES
                DEVUELVE TRIO
            SINO
                DEVUELVE FULL
            FINSI
        FINSI
    FINSI
FINSI
FINSI
```

### 3.5.2. Diagrama o autómeta

A continuación podemos ver el diagrama que anteriormente enseñamos en forma de algoritmo.

Este primer diagrama evalúa si la combinación de manos del jugador, contando las cartas comunitarias más las personales, tienen cinco cartas del mismo color o forman escalera. Si sucede alguna de estas combinaciones, no puede pasar el resto.



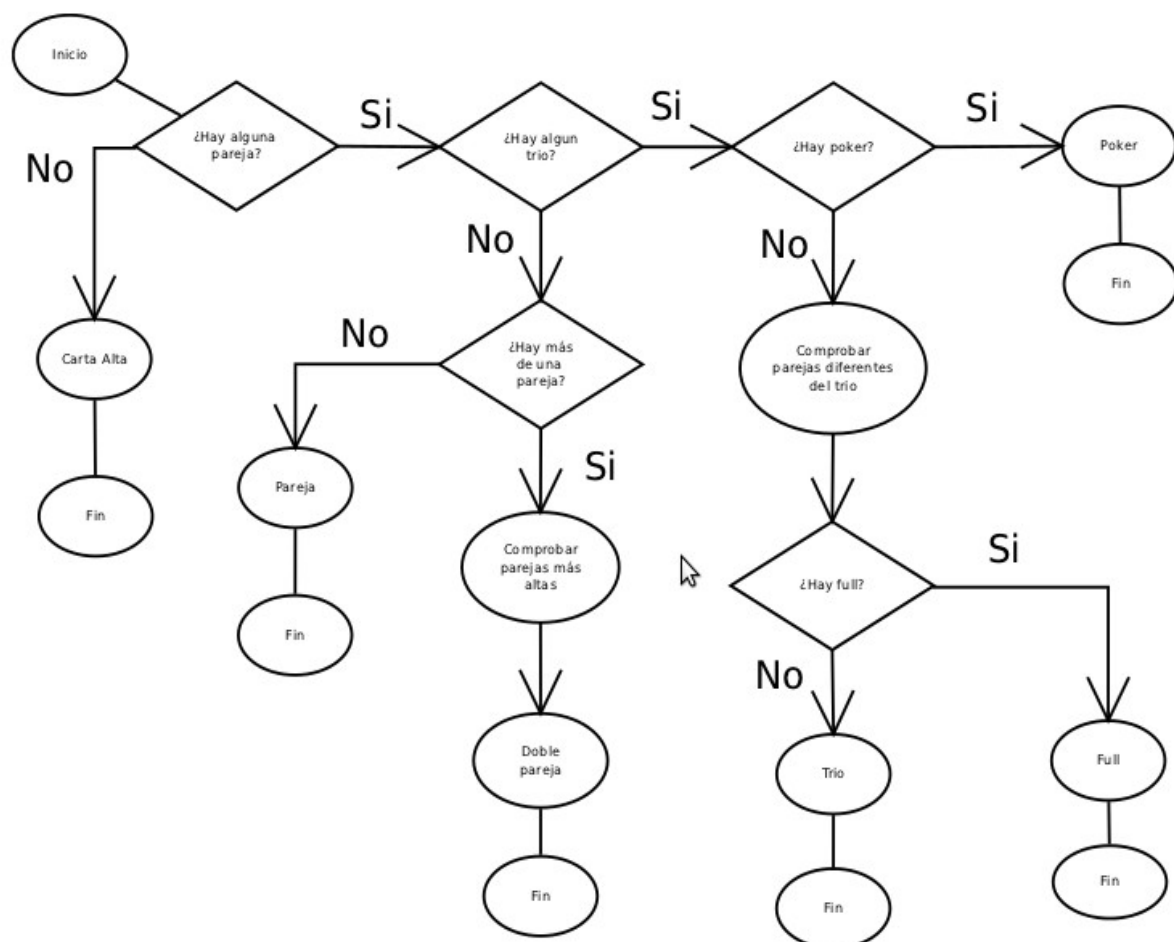
15. Figura: Diagrama de formación de manos, Escaleras.

El diagrama comienza comprobando si hay cinco cartas del mismo palo. Si hay cinco cartas del mismo palo, pasa a la siguiente fase del diagrama en la parte izquierda del mismo. En esta comprobación mira si hay cinco cartas consecutivas que formen escalera. Si no hay escalera se acaba la comprobación y el diagrama decide que hay color. Al continuar por la otra posibilidad, que si haya escalera, tiene que comprobar si es escalera

real o solamente de color. La diferencia entre las dos es que la escalera real acaba con un As en la parte alta de la mano, la combinación sería la siguiente: Diez, jota, reina, rey y as.

Al principio del diagrama comprobamos si habían cinco cartas del mismo palo. Ahora observamos la parte en que no las hay, esto lleva a comprobar si hay escalera:

- Si la hay, se para la ejecución del diagrama.
- Si no hay escalera, tendríamos que continuar con el siguiente diagrama de evaluación de manos.



16. Figura: Diagrama de formación de manos.

En este diagrama se evalúan las posibles manos que puede tener un jugador determinado, no comprobando las escaleras ni el color ya que se comprobaron anteriormente.

## 4. Multi-jugador offline

En este apartado mostraremos el apartado multi-jugador offline, en el que no solo hablaremos del juego en si, si no que también mostraremos la interfaz creada para esta versión del juego.

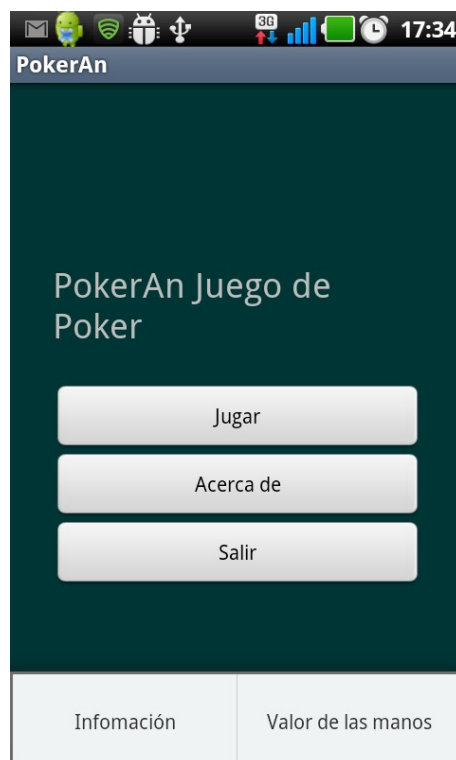
La peculiaridad de este modo, a diferencia de los disponibles para el dispositivo, es poder jugar con más de un jugador en un mismo dispositivo, creando una forma fácil y rápida de jugar al poker, con la única necesidad de un dispositivo con Android y más de un jugador.

El propio juego se encarga de hacer los cambios de jugador oportunos, esperando al jugador para comenzar su apuesta.

### 4.2. Interfaz del juego

La interfaz de este juego se distribuye en vistas, donde cada vista tiene un controlador en el que se ejecutan métodos invisibles para los usuarios y también se llaman a otras vistas para poder cambiar la funcionalidad visible para el jugador.

#### Pantalla principal

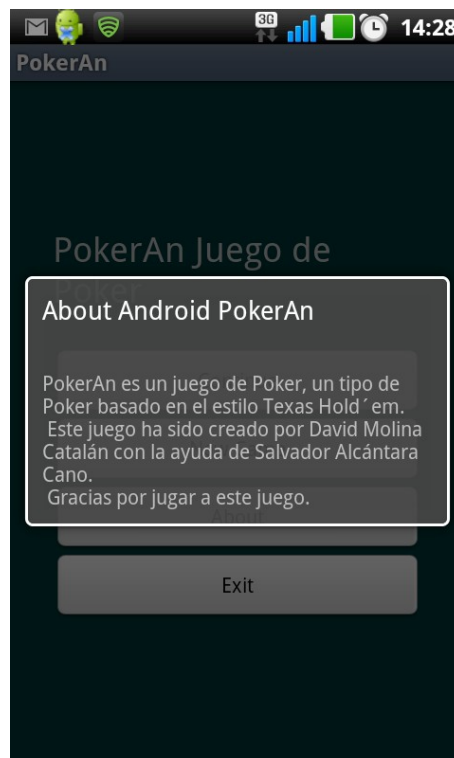


17. Figura: Pantalla principal.

En esta pantalla podemos ver que hay tres opciones:

- Jugar: este botón da opción a comenzar una partida nueva, lo cual al iniciarla borra automáticamente la posible partida guardada. Esta acción da lugar a la creación de jugadores, con la posibilidad de crear de dos a ocho usuarios. Esta vista carga la vista y las opciones para la creación de jugadores.
- Acerca de: al pulsar esta opción únicamente se iniciará una Activity (vista en Android) encargada de mostrar un texto de agradecimientos, versión y datos acerca de la aplicación.
- Salir: esta es la única opción que no carga ninguna vista. Su única funcionalidad es salir de la aplicación.
- Menú inferior: en el menú inferior hay dos posibles opciones para ver información sobre el poker:
  - Información: en este botón se muestra información básica sobre las reglas del juego.
  - Valor de las manos: se muestra el valor de las manos según sus combinaciones.

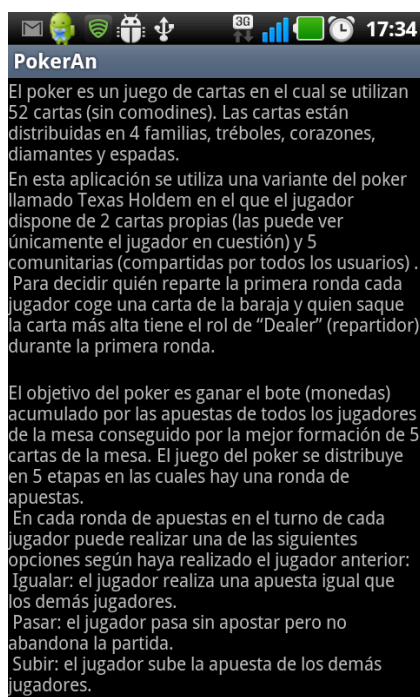
## Pantalla Acerca de



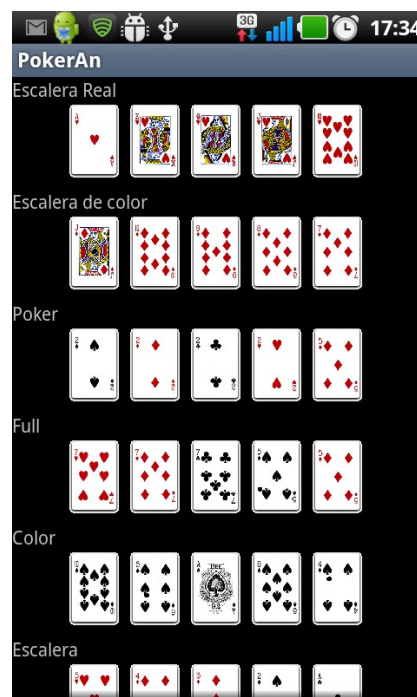
18. Figura: Pantalla Acerca de.

Como indicamos anteriormente esta pantalla únicamente sirve para los agradecimientos.

## Pantallas información y valor de las manos



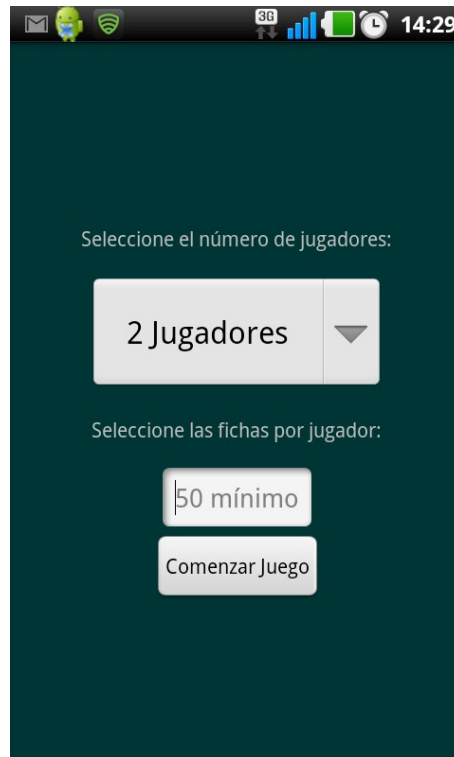
19. Figura: Pantallas de Información.



20. Figura: Valor de las manos

En esta imagen se muestra a la izquierda la pantalla de información donde se indican las reglas básicas del juego y en la imagen de la derecha se muestran las combinaciones de cartas y el orden de su valor: de arriba a bajo y de mayor a menor.

## Pantalla de creación de partida



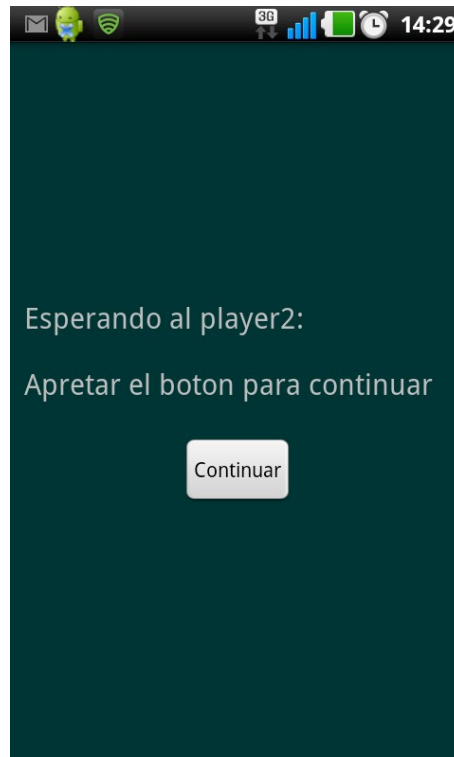
21. Figura: Pantalla de creación de partida.

Esta vista se encarga de la creación de la partida seleccionando de la lista el número de jugadores: entre 2 a 6. También se puede elegir las fichas de cada jugador, como mínimo 50 que es la cantidad que ejecuta por defecto, para tener una experiencia atractiva del juego.

En esta pantalla internamente se encarga de inicializar las variables y de dar paso al controlador de la partida y su vista por defecto.



## Pantalla del mediador o espera del jugador



22. Figura: Pantalla del mediador o espera del jugador.

En esta vista como podemos ver es la pantalla de espera del jugador.

Como funcionalidades para el usuario solo tiene esperar al jugador indicado para comenzar su turno y poder realizar la apuesta y ver las cartas, tanto propias como comunitarias.

Al apretar el botón, esta vista dará paso a la pantalla del jugador. Si es el comienzo del turno esta pantalla ejecutara la vista de jugador donde se ejecutan las apuestas obligatorias y si ya han pasado las apuestas obligatorias dará paso a la pantalla de apuestas del jugador.

## Pantalla de apuesta obligatoria



23. Figura: Pantalla de apuesta obligatoria.

En esta pantalla se muestran los jugadores que tienen que apostar obligatoriamente. La ciega pequeña apuesta 5 monedas y la ciega grande el doble. Al apretar el botón dará paso a la pantalla del mediador.

Internamente el controlador de esta pantalla descuenta las apuestas del jugador de sus monedas y envía al mediador del juego quién es el siguiente jugador activo para continuar la partida.

## Pantalla de individual del jugador



24. Figura: Pantallas individual del jugador.



25. Figura: Pantallas individual del jugador.

En la anterior imagen se observan dos pantallas que pertenecen a la misma vista. La vista es la pantalla individual y privada del jugador, donde la pantalla de la izquierda se muestran con 2 jugadores y la de la derecha con 6.

Como podemos ver al principio de la pantalla se indica el jugador al que pertenece el turno. Seguidamente vemos la fase en que se encuentra, en este caso “preflop”.

Después de la fase donde esta, se indican cada uno de los jugadores, con la las monedas de cada jugador y la apuesta acumulada por cada uno de ellos. Encima de las cartas se puede ver las monedas acumuladas de la mesa y las acumuladas por el turno.

Las cartas que se pueden observar son siete, cinco comunitarias (parte superior) y las dos individuales (parte inferior). Las cartas no tienen ninguna acción propia, simplemente indicar al jugador sus cartas y valor.

Debajo de las cartas se muestran las acciones disponibles del jugador con sus botones correspondientes. Los botones son dinámicos, solo se muestran aquellos a los que el jugador puede acceder.

Al apretar uno de los botones, el controlador realiza la acción comunicándolo al modelo guardando las variables oportunas y llamando al controlador/mediador del juego para esperar al siguiente jugador.

En la siguiente imagen se puede ver la misma vista en distintos turnos, con los cambios de las cartas, nombres y botones que se pueden observar.



26. Figura: Pantallas con cambio de turno.

## Pantalla del ganador



27. Figura: Pantalla del ganador del turno.



28. Figura: Pantalla del ganador de la partida.

La pantalla de la izquierda muestra la vista donde indica el ganador del turno y al apretar el botón vuelve al controlador al turno inicial.

La pantalla de la derecha muestra el ganador de la partida y al apretar inicio vuela al inicio de la aplicación, para poder volver a jugar de nuevo.

## 6. Conclusión

En cuanto a las conclusiones y el trabajo realizado en el proyecto, comenzaremos hablando sobre los objetivos planteados en el estudio de viabilidad, en el que como objetivo principal estaba y sigue estando el aprendizaje.

Este objetivo se ha cumplido al haber adquirido unos conocimientos antes no desarrollados, como pueden ser el uso de patrones de diseño (muy importantes para cualquier tipo de desarrollo software). Otro punto fuerte ha sido profundizar en Android que aunque no sea un lenguaje, propiamente dicho, tiene un estilo diferente y propio para ser programado.

Otro punto interesante a nombrar es la importancia de la planificación, ha sido más difícil de llevar a cabo al tener que compaginar varias actividades diferente al proyecto.

Con lo anterior comentado y la consecución de la aplicación, se quiere dar un punto de vista diferente del poker usándolo desde un dispositivo móvil. De esta forma las personas interesadas pueden aprender con la ayuda del programa o simplemente divertirse jugando, acompañados de algún amigo o conocido.

La aplicación desarrollada se convierte así en un medio de aprendizaje o simplemente de ocio, divertido y ameno, disponible para todo aquel que quiera adquirirlo de forma completamente gratuita.

### ***6.1. Desviaciones de la planificación y problemas encontrados***

Con respecto a las desviaciones en la planificación, el tiempo se alargó en la implementación de la aplicación, al adaptar el código Java del modelo para poder hacerlo funcionar correctamente en Android.

Otro retraso se produjo también en el diseño y creación de las vistas. En la creación de las vistas el problema ha sido desconocer el lenguaje en que se desarrollaba la interfaz gráfica de Android (XML), una vez utilizada se puede comprobar que alberga muchas posibilidades.

El problema comentado fue que al realizar la aplicación primero en Java para comprobar que el modelo funcionaba correctamente, el patrón mediador estaba programado de tal forma que Android no lo podía ejecutar. Debe ser el controlador del patrón MVC quien haga de mediador y así utilice su ejecución para controlar el modelo y las vistas.

Otro problema a mencionar ha sido la no implementación del bluetooth en el juego por falta de tiempo, pero que al acabar el proyecto final se seguirá trabajando en él para acabar esta función y ampliar el juego con algunas de las mejoras que nombraremos en el siguiente apartado.

## **6.2. Ampliaciones y mejoras**

En esta sección se plantean las propuestas de mejoras que se pueden aplicar al proyecto en un futuro.

Como mejora principal se plantea añadir funcionalidad para poder crear partidas online conectadas directamente con las personas con las que se quiera participar. También se contempla crear partidas públicas para que así las personas que lo deseen jugar también con desconocidos.

En la siguiente mejora se pretende añadir otros tipos de juego de cartas, relacionados en un principio con el poker, pero que en un futuro se puedan jugar a otros tipos de juego también popularizados.

Por ultimo y como mejora a nivel de comunicación nos gustaría implementar un chat público en las salas de juego públicas. Una vez se haya incluido esta función, también la valorará la posibilidad de enviar mensajes privados a usuarios registrados. Con esta ultima mejora habría que crear un servidor y una base de datos acorde a lo necesitado.

Como se puede comprobar, el proyecto se puede ampliar según las necesidades y los usuarios a los que se quiera llegar, dando unas opciones prácticamente ilimitadas en el juego de cartas.

## **6.3. Valoración personal**

La elaboración de este proyecto me ha supuesto un gran aliciente tanto a nivel personal como profesional. He profundizado en un lenguaje poco utilizado en la carrera, hasta este ultimo año, como es Java, en el que se ha indagado más. Además de por el uso del lenguaje Java, el haber escogido Android como plataforma móvil para el proyecto final facilita las cosas para el acceso al mundo laboral al estar este socialmente muy valorado, utilizándolo cada vez más personas en su vida cotidiana.

La temática de este proyecto era una aplicación para que otras personas pudieran disfrutar de ella, sin perder de vista que el objetivo principal era aprender más sobre la programación, entender más la tecnología móvil, demostrando las posibilidades que puede llegar a tener, y acercarse más a la planificación de un proyecto que, aún estando dirigido por un profesor, está enfocado a simular cualquier situación en el mundo laboral a la que nos podamos enfrentar.

Este proyecto me ha dado la posibilidad de ver la dificultad y el tiempo de dedicación que puede llegar a tener un proyecto creado para dispositivos móviles. Una cosa importante que me ha ayudado a progresar ha sido ir solucionando los problemas e inconvenientes que iban surgiendo a medida que se desarrollaba el proyecto.

Personalmente estoy muy satisfecho con el resultado del proyecto, con todo lo que he aprendido y por haberme dado cuenta de las posibilidades que ofrece esta ingeniería de cara al mundo laboral.



## 7. Bibliografía

A continuación se detallaran todas las fuentes de información que se han utilizado para el desarrollo de nuestro proyecto.

### Libros consultados

Frank Ableson, Charlie Collins y Robi Sen. **“Android, guía para desarrolladores”** . Editorial Anaya Multimedia. Año 2009.

Bruce Eckel. **“Piensa en Java”**. Editorial Prentice Hall. Año 2003.

Perdita Stevens y Rob Pooley. **“Utilización de UML”**. Editorial Pearson. Año 2007.

### Enlaces consultados

Video-tutoriales. [<http://www.edu4java.com/>]. Última consulta Abril 2012.

Blog de programación en Android. [<http://www.sgoliver.net/blog/>]. Última consulta Mayo 2012.

Web de tutoriales Android. [<http://www.tutorialandroid.com/>]. Última consulta Mayo 2012.

Web oficial de desarrollo en Android. [<http://developer.android.com/index.html>]. Última consulta Junio 2012.

Web oficial de la Api de Java [<http://docs.oracle.com/javase/1.5.0/docs/api/>]. Última consulta Junio 2012.